Neural Analogical Matching

Maxwell Crouse[1]*, **Constantine Nakos**[1], **Ibrahim Abdelaziz**[2], **Ken Forbus**[1]
[1]Qualitative Reasoning Group, Northwestern University
[2]IBM Research, IBM T.J. Watson Research Center
{mvcrouse, cnakos}@u.northwestern.edu, ibrahim.abdelaziz1@ibm.com, forbus@northwestern.edu

Abstract

Analogy is core to human cognition. It allows us to solve problems based on prior experience, it governs the way we conceptualize new information, and it even influences our visual perception. The importance of analogy to humans has made it an active area of research in the broader field of artificial intelligence, resulting in data-efficient models that learn and reason in human-like ways. While cognitive perspectives of analogy and deep learning have generally been studied independently of one another, the integration of the two lines of research is a promising step towards more robust and efficient learning techniques. As part of a growing body of research on such an integration, we introduce the Analogical Matching Network: a neural architecture that learns to produce analogies between structured, symbolic representations that are largely consistent with the principles of Structure-Mapping Theory.

1 Introduction

Analogical reasoning is a form of inductive reasoning that cognitive scientists consider to be one of the cornerstones of human intelligence (Gentner 2003; Hofstadter 2001, 1995). Analogy shows up at nearly every level of human cognition, from low-level visual processing (Sagi, Gentner, and Lovett 2012) to abstract conceptual change (Gentner et al. 1997). Problem solving using analogy is common, with past solutions forming the basis for dealing with new problems (Holyoak, Junn, and Billman 1984; Novick 1988). Analogy also facilitates learning and understanding by allowing people to generalize specific situations into increasingly abstract schemas (Gick and Holyoak 1983).

Many different theories have been proposed for how humans perform analogy (Mitchell 1993; Chalmers, French, and Hofstadter 1992; Gentner 1983; Holyoak, Holyoak, and Thagard 1995). One of the most influential theories is Structure-Mapping Theory (SMT) (Gentner 1983), which posits that analogy involves the alignment of structured representations of objects or situations subject to certain constraints. Key characteristics of SMT are its use of symbolic representations and its emphasis on relational structure, which allow the same principles to apply to a wide variety of domains.

*Correspondence to mvcrouse@u.northwestern.edu

Until now, the symbolic, structured nature of SMT has made it a poor fit for deep learning. The representations produced by deep learning techniques are incompatible with off-the-shelf SMT implementations like the Structure-Mapping Engine (SME) (Falkenhainer, Forbus, and Gentner 1989; Forbus et al. 2017), while the symbolic graphs that SMT assumes as input are challenging to encode with traditional neural methods. In this work, we describe how recent advances in graph representation learning can be leveraged to create deep learning systems that can learn to produce structural analogies consistent with SMT.

Contributions: We introduce the Analogical Matching Network (AMN), a neural architecture that learns to produce analogies between symbolic representations. AMN is trained on purely synthetic data and is demonstrated over a diverse set of analogy problems drawn from structure-mapping literature to produce outputs that are largely consistent with SMT. With AMN, we aim to push the boundaries of deep learning and extend them to an important area of human cognition; in particular, by showing how to design a deep learning system that conforms to a cognitive theory of analogical reasoning. It is our hope that future generations of neural architectures can reap the same benefits from analogy that symbolic reasoning systems and humans currently do.

2 Related Work

Many different computational models of analogy have been proposed (Mitchell 1993; Holyoak and Thagard 1989; O'Donoghue and Keane 1999; Forbus et al. 2017), each instantiating a different cognitive theory of analogy. The differences between them are compounded by the computational costs of analogical reasoning, a provably NP-Hard problem (Veale and Keane 1997). While these computational models are often used to test cognitive theories of human behavior, they are also useful tools for applied tasks. For instance, the Structure-Mapping Engine (SME) has been used in question-answering (Ribeiro et al. 2019), computer vision (Chen et al. 2019), and machine reasoning (Klenk et al. 2005).

Many of the early approaches to analogy were connectionist (Gentner and Markman 1993). The STAR architecture of (Halford et al. 1994) used tensor product representations of structured data to perform simple analogies of the form $R(x, y) \Rightarrow S(f(x), f(y))$. Drama (Eliasmith and Thagard

[1] nucleus	[8] sun
[2] electron	[9] planet
[3] MASS([1])	[10] MASS([8])
[4] MASS([2])	[11] MASS([9])
[5] ATTRACTS([1]], [2])	[12] TEMPERATURE([8]])
[6] REVOLVES-AROUND([2], [1])	[13] TEMPERATURE([9]])
[7] GREATER([3], [4])	[14] REVOLVES-AROUND([9], [8])
	[15] GREATER([10], [11])
	[16] GREATER([12], [13])
	[17] ATTRACTS([9], [8])
	[18] CAUSES(AND([15], [17]), [14])
	[19] YELLOW([8]])

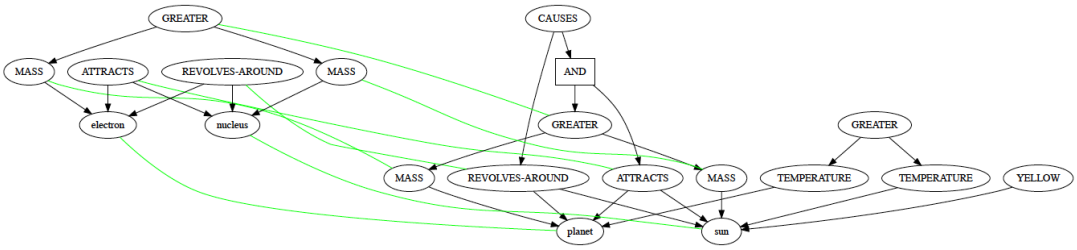

Figure 1: Relational and graph representations for models of the atom (left) and Solar System (right). Light green edges indicate the set of correspondences between the two graphs.

2001) was an implementation of the multi-constraint theory of analogy (Holyoak, Holyoak, and Thagard 1995) that used holographic representations similar to tensor products to embed structure. LISA (Hummel and Holyoak 1997, 2005) was a hybrid symbolic connectionist approach to analogy. It staged the mapping process temporally, generating mappings from elements that were activated at the same time.

Cognitive perspectives of analogy have gone relatively unexplored in deep learning research, with only a few recent works that address them (Hill et al. 2019; Zhang et al. 2019; Lu et al. 2019). Most prior deep learning works have considered analogies involving perceptual data (Mikolov, Yih, and Zweig 2013; Reed et al. 2015; Bojanowski et al. 2017; Zhou et al. 2019; Benaim et al. 2020). Such problems differ from those seen in the structure-mapping literature in that they typically do not require explicit graph matching and they involve only one relation which is unobserved.

Our approach is conceptually related to recent work on neural graph matching (Emami and Ranka 2018; Georgiev and Lió 2020; Wang, Yan, and Yang 2019). Such works generally focus on finding unconstrained maximum weight matchings and often interleave their networks with hard-coded algorithms (e.g., (Emami and Ranka 2018) applies the Hungarian algorithm to coerce its outputs into a permutation matrix). These considerations make them less applicable here, as 1) SMT is subject to unique constraints that make standard bipartite matching techniques insufficient and 2) we wish to explore the extent to which SMT is purely learnable.

3 Structure-Mapping Theory

In Structure-Mapping Theory (SMT) (Gentner 1983), analogy centers around the structural alignment of *relational representations* (see Figure 1). A relational representation is a set of logical expressions constructed from entities (e.g., `sun`), attributes (e.g., `YELLOW`), functions (e.g., `TEMPERATURE`), and relations (e.g., `GREATER`). Structural alignment is the process of producing a *mapping* between two relational representations (referred to as the *base* and *target*). A mapping is a triple $\langle M, C, S \rangle$, where M is a set of *correspondences* between the base and target, C is a set of *candidate inferences* (i.e., inferences about the target that can be made from the structure of the base), and S is a *structural evaluation score* that measures the quality of M. Correspondences are pairs of elements between the base and target (i.e., expressions or entities) that are identified as matching with one another. While entities can be matched together irrespective of their labels, there are more rigorous criteria for matching expressions. SMT asserts that matches should satisfy the following:

1. *One-to-One*: Each element of the base and target can be a part of *at most* one correspondence.
2. *Parallel Connectivity*: Two expressions can be in a correspondence with each other only if their arguments are also in correspondences with each other.
3. *Tiered Identicality*: Relations of expressions in a correspondence must match identically, but functions need not if their correspondence supports parallel connectivity.
4. *Systematicity*: Preference should be given to mappings with more deeply nested expressions.

To understand these properties, we use a classic analogy (see Figure 1) from (Gentner 1983; Falkenhainer, Forbus, and Gentner 1989), which draws an analogy between the Solar System and the Rutherford model of the atom. A set of correspondences M between the base (Solar System) and target (Rutherford atom) is a set of pairs of elements from both sets, e.g., $\{\langle$ `[1], [8]` $\rangle, \langle$ `[2], [9]` $\rangle\}$. The one-to-one constraint

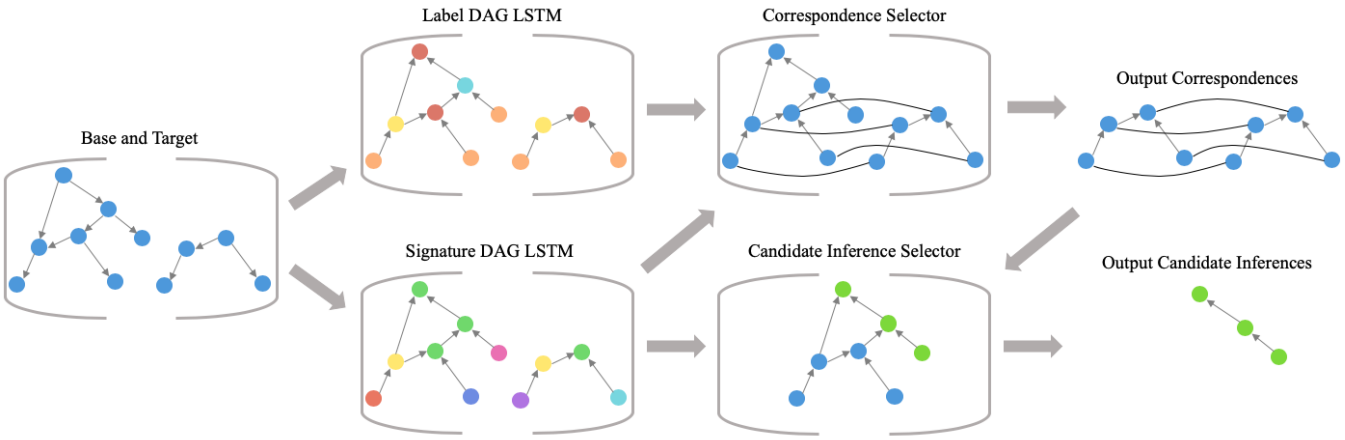

Figure 2: An overview of the model pipeline

restricts each element to be a member of at most one correspondence. Thus, if $\langle$ `[7], [15]` $\rangle$ was a member of M, then $\langle$ `[7], [16]` $\rangle$ could not be added to M. Parallel connectivity enforces correspondence between arguments if the parents are in correspondence. In this example, if $\langle$ `[7], [15]` $\rangle$ was a member of M, then both $\langle$ `[3], [10]` $\rangle$ and $\langle$ `[4], [11]` $\rangle$ would need to be members of M. Parallel connectivity also respects argument order when dealing with ordered relations. Tiered identicality is not relevant in this example; however, if `[10]` used the label `WEIGHT` instead of `MASS`, tiered identicality could be used to match `[3]` and `[10]`, since such a correspondence would allow for a match between their parents. The last property, systematicity, results in larger correspondence sets being preferred over smaller ones. Note that the singleton set $\{\langle$ `[1], [8]` $\rangle\}$ satisfies SMT's constraints, but it is clearly not useful by itself. Systematicity captures the natural preference for larger, more interesting matches.

Candidate inferences are statements from the base that are projected into the target to fill in missing structure (Bowdle and Gentner 1997; Gentner and Markman 1998). Given a set of correspondences M, candidate inferences are created from statements in the base that are supported by expressions in M but are not part of M themselves. In Figure 1, one candidate inference would be `CAUSES(AND([7],[5]),[6])`, derived from `[18]` by substituting its arguments with the expressions they correspond to in the target. In this work, we adopt SME's default criteria for computing candidate inferences. Valid candidate inferences are all statements that have *some* dependency that is included in the correspondences or an ancestor that is a candidate inference (e.g., an expression whose parent has arguments in the correspondences).

The concepts above carry over naturally into graph-theoretic notions. The base and target are considered semi-ordered directed-acyclic graphs (DAGs) $G_B = \langle V_B, E_B \rangle$ and $G_T = \langle V_T, E_T \rangle$, where V_B and V_T are sets of nodes and E_B and E_T are sets of edges. Each node corresponds to some expression and has a label given by its relation, function, attribute, or entity name. Structural alignment is the process of finding a maximum weight bipartite matching $M \subseteq V_B \times V_T$, where M satisfies the pairwise-disjunctive constraints imposed by parallel connectivity. Finding candidate inferences is then determining the subset of nodes from $V_B \setminus \{b_i : \langle b_i, t_j \rangle \in M\}$ with support in M.

4 Model

4.1 Model Components

Given a base $G_B = \langle V_B, E_B \rangle$ and target $G_T = \langle V_T, E_T \rangle$, AMN produces a set of correspondences $M \subseteq V_B \times V_T$ and a set of candidate inferences $I \in V_B \setminus \{b_i : \langle b_i, t_j \rangle \in M\}$. A key design choice of this work was to avoid using rules or architectures that force particular outputs whenever possible. AMN is *not* forced to output correspondences that satisfy the constraints of SMT; instead, conformance with SMT is reinforced through performance on training data. Our architecture uses Transformers (Vaswani et al. 2017) and pointer networks (Vinyals, Fortunato, and Jaitly 2015) and takes inspiration from the work of (Kool, Van Hoof, and Welling 2018). A high-level overview is given in Figure 2, which shows how each of the three main components (graph embedding, correspondence selection, and candidate inference selection) interact with one another.

Representing Structure: When embedding the nodes of G_B and G_T, there are representational concerns to keep in mind. First, as matching should be done on the basis of structure, the labels of entities should not be taken into account during the alignment process. Second, because SMT's constraints require AMN to be able to recognize when a node is part of multiple correspondences, AMN should maintain distinguishable representations for distinct nodes, even if those nodes have the same labels. Last, the architecture should not be vocabulary dependent, i.e., AMN should generalize to symbols it has never seen before. To achieve each of these, AMN first parses the original input into two separate graphs, a *label graph* and a *signature graph* (see Figure 3).

The label graph will be used to get an estimate of structural similarities. To generate the label graph, AMN substitutes each entity node's label with a generic entity token. This is

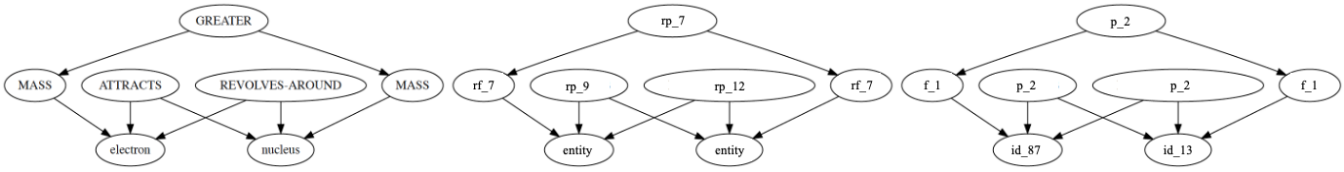

Figure 3: Original graph (left), its label graph (middle), and its signature graph (right)

intentional, as it reflects that entity labels have no inherent utility for producing matchings according to SMT. Then, each function and predicate node is assigned a randomly chosen generic label (from a fixed set of such labels) based off its arity and orderedness. Assignments are made consistently across the entire graph, e.g., *every* instance of `MASS` in *both* the base and target would be assigned the same generic replacement label. This substitution means the original label is not used in the matching process, which allows AMN to generalize to new symbols.

The label graph is not sufficient to produce representations that can be used for matching, as it represents a node by only label-based features which are shared amongst different nodes, an issue known as the *type-token distinction* (Kahneman, Treisman, and Gibbs 1992; Wetzel 2009). To contend with this, a signature graph is constructed that represents nodes in a way that respects object identity. To construct the signature graph, AMN replaces each distinct entity with a unique identifier (drawn from a fixed set of possible identifiers). It then assigns each function and predicate a new label based solely on its arity and orderedness, ignoring the original symbol. For instance, `ATTRACTS` and `REVOLVES-AROUND` would be assigned the same label as they are both ordered binary predicates.

As all input graphs will be DAGs, AMN uses two separate DAG LSTMs (Crouse et al. 2019) to embed the nodes of the label and signature graphs (equations detailed in Appendix 7.4). Each node embedding is computed as a function of its complete set of dependencies in the original graph. The set of label structure embeddings is written as $L_V = \{l_v : v \in V\}$ and the set of signature embeddings is written as $S_V = \{s_v : v \in V\}$. Before passing these embeddings to the next step, each element of S_V is scaled to unit length, i.e. each s_v becomes $s_v / \|s_v\|$, which gives our network an efficiently checkable criterion for whether or not two nodes are likely to be equal, i.e., when the dot product of two signature embeddings is 1.

Correspondence Selector: The graph embedding procedure yields two sets of node embeddings (label structure and signature embeddings) for the base and target. We utilize the set of embedding pairs for each node of V_B and V_T, writing l_v to denote the label structure embedding of node v from L_V and s_v the signature embedding of node v from S_V. We first define the set of unprocessed correspondences $\mathcal{C}^{(0)}$

$$\hat{\mathcal{C}} = \{\langle b, t\rangle \in V_B \times V_T : \|l_b - l_t\| \leq \epsilon\}$$
$$\mathcal{C}^{(0)} = \{\langle [l_b; l_t; s_b; s_t], s_b, s_t\rangle : \langle b, t\rangle \in \hat{\mathcal{C}}\}$$

where $[\cdot;\cdot]$ denotes vector concatenation, ϵ is the tiered identicality threshold that governs how much the subgraphs rooted at two nodes may differ and still be considered for correspondence (in this work, we set $\epsilon = 1\text{e}{-5}$). The first element of each correspondence in $\mathcal{C}^{(0)}$, i.e., $h_c = [l_b; l_t; s_b; s_t]$, is then passed through an N-layered Transformer encoder (equations detailed in Appendix 7.4) to produce a set of encoded correspondences

$$\mathcal{E} = \{\langle h_c^{(N)}, s_b, s_t\rangle \in \mathcal{C}^{(N)}\}$$

The Transformer decoder selects a subset of correspondences that constitutes the best analogical match (see Figure 4). The attention-based transformations are only performed on the initial element of each tuple, i.e., h_d in $\langle h_d, s_b, s_t\rangle$. We let $\mathcal{D}_t$ be the processed set of all selected correspondences at timestep t (after the N attention layers) and $\mathcal{O}_t$ be the set of all remaining correspondences (with $\mathcal{D}_0 = \{\texttt{START-TOK}\}$ and $\mathcal{O}_0 = \mathcal{E} \cup \{\texttt{END-TOK}\}$). The decoder generates compatibility scores α_{od} between each pair of elements, i.e., $\langle o, d\rangle \in \mathcal{O}_t \times \mathcal{D}_t$. These are combined with the signature embedding similarities to produce a final compatibility π_{od}

$$\pi_{od} = \text{FFN}\big([\tanh(\alpha_{od}); s_{b_o}^\top s_{b_d}; s_{t_o}^\top s_{t_d}]\big)$$

where FFN is a two layer feed-forward network with ELU activations (Clevert, Unterthiner, and Hochreiter 2016). Recall that the signature components, i.e. s_b and s_t, were scaled to unit length. Thus, we would expect closeness in the original graph to be reflected by dot-product similarity and identicality to be indicated by a maximum value dot-product, i.e. $s_{b_o}^\top s_{b_d} = 1$ or $s_{t_o}^\top s_{t_d} = 1$. Once each pair has been scored, AMN selects an element of $\mathcal{O}_t$ to be added to $\mathcal{D}_{t+1}$. For each $o \in \mathcal{O}_t$, we compute its value to be

$$v_o = \text{FFN}\big([\max_d \pi_{od}; \min_d \pi_{od}; \sum_d \frac{\pi_{od}}{|\mathcal{D}_t|}]\big)$$

where FFN is a two layer feed-forward network with ELU activations. A softmax is applied to these scores and the highest valued element is added to $\mathcal{D}_{t+1}$. The use of maximum, minimum, and average is intended to let the network capture both individual and aggregate evidence. Individual evidence is given by a pairwise interaction between two correspondences (e.g., two correspondences that together violate the one-to-one constraint). Conversely, aggregate evidence is given by the interaction of a correspondence with everything selected thus far (e.g., a correspondence needed for several parallel connectivity constraints). When `END-TOK` is selected, the set of correspondences M returned is the set of node pairs from V_B and V_T associated with elements in $\mathcal{D}$.

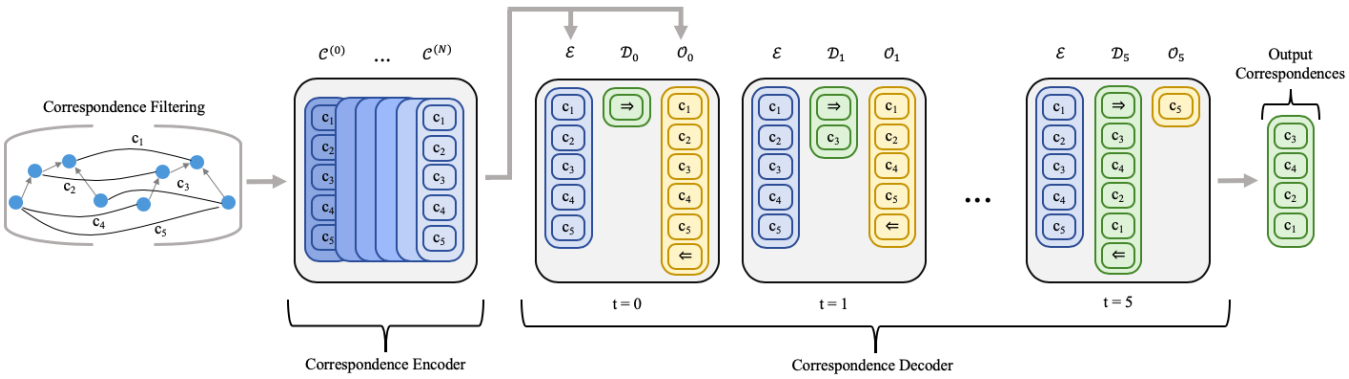

Figure 4: The correspondence selection process, where $\Rightarrow$ and $\Leftarrow$ are the start and stop tokens and $\mathcal{E}$, $\mathcal{D}_t$, and $\mathcal{O}_t$ are the sets of encoded, selected, and remaining correspondences

Candidate Inference Selector: The output of the correspondence selector is a set of correspondences M. The candidate inferences associated with M are drawn from the nodes of the base graph V_B that were *not* used in M. Let V_{in} and V_{out} be the subsets of V_B that were / were not used in M, respectively. We first extract all signature embeddings for both sets, i.e., $\mathcal{S}_{in} = \{s_b : b \in V_{in}\}$ and $\mathcal{S}_{out} = \{s_b : b \in V_{out}\}$. In this module there are no Transformer components, with AMN operating directly on $\mathcal{S}_{in}$ and $\mathcal{S}_{out}$.

AMN will select elements from $\mathcal{S}_{out}$ to return. Like before, we let $\mathcal{D}_t$ be the set of all selected elements from $\mathcal{S}_{out}$ and $\mathcal{O}_t$ be the set of all remaining elements from $\mathcal{S}_{out}$ at timestep t. AMN computes compatibility scores between pairs of output options with candidate inference and previously selected nodes, i.e. α_{od} for each $\langle o, d \rangle \in \mathcal{O}_t \times (\mathcal{D}_t \cup \mathcal{S}_{in})$. The compatibility scores are given by a simple single-headed attention computation (see Appendix 7.4). Unlike the correspondence encoder-decoder, there are no other values to combine these scores with, so they are used directly to compute a value v_o for each element of $\mathcal{O}_t$. AMN computes this value as

$$\alpha'_{od} = \tanh(\alpha_{od})$$

$$v_o = \text{FFN}\Big(\Big[\max_d \alpha'_{od}; \min_d \alpha'_{od}; \sum_d \frac{\alpha'_{od}}{|\mathcal{D}_t|}\Big]\Big)$$

A softmax is used and the highest valued element is added to $\mathcal{D}_{t+1}$. Once the end token is selected, decoding stops and the set of nodes associated with elements in $\mathcal{D}$ is returned.

Loss Function: As both the correspondence and candidate inference components use a softmax, the loss function is categorical cross entropy. Teacher forcing is used to guide the decoder to select the correct choices during training. With $\mathcal{L}_{corr}$ the loss for correspondence selection and $\mathcal{L}_{ci}$ the loss for candidate inference selection, the final loss is given as $\mathcal{L} = \mathcal{L}_{corr} + \lambda\mathcal{L}_{ci}$ (with λ a hyperparameter), which is minimized with Adam (Kingma and Ba 2014).

4.2 Model Scoring

Structural Match Scoring: In order to avoid counting erroneous correspondence predictions towards the score of the output correspondences M, we first identify all correspondences that are either degenerate or violate the constraints of SMT. Degenerate correspondences are correspondences between constants that have no higher-order structural support in M (i.e., if either has no parent that participates in a correspondence in M). To determine if a correspondence $\langle b, t \rangle$ violates SMT, we check whether the subgraphs of the base and target rooted at b and t satisfy the one-to-one matching, parallel connectivity, and tiered identicality constraints (see Section 3). The check can be computed in time linear with the size of the corresponding subgraphs. Let the valid subset of M be M_{val}. A correspondence m is considered a *root correspondence* if there does not exist another correspondence m' such that $m' \in M_{val}$ and a node in m' is an ancestor of a node in m. We define $M_{root} \subseteq M_{val}$ to be the set of all such root correspondences. For a correspondence $m = \langle b, t \rangle$ in M_{val}, its score $s(m)$ is given as the size of the subgraph rooted at b in the base. The structural match score for M is then sum of scores for all correspondences in M_{root}, i.e., $s(M) = \sum_{m \in M_{root}} s(m)$. This repeatedly counts nodes that appear in the dependencies of multiple correspondences, which leads to higher scores for more interconnected matchings (in keeping with the systematicity preference of SMT).

Structural Evaluation Maximization: Dynamically assigning labels to each example allows AMN to handle never-before-seen symbols, but its inherent randomness can lead to significant variability in terms of outputs. AMN combats this by running each test problem r times and returning the mapping $M = \arg\max_{M_i} \sum_j J(M_i, M_j)$, where $J(M_i, M_j)$ is the Jaccard index (intersection over union) between the correspondence sets produced by the i-th and j-th runs. Intuitively, this is the run that shared the most correspondences with other runs and had the fewest unshared extra correspondences.

5 Experiments

5.1 Data Generation and Training

AMN was trained on 100,000 synthetic analogy examples, with the hyperparameters used for AMN provided in Appendix 7.1 (in the supplementary material). A single example

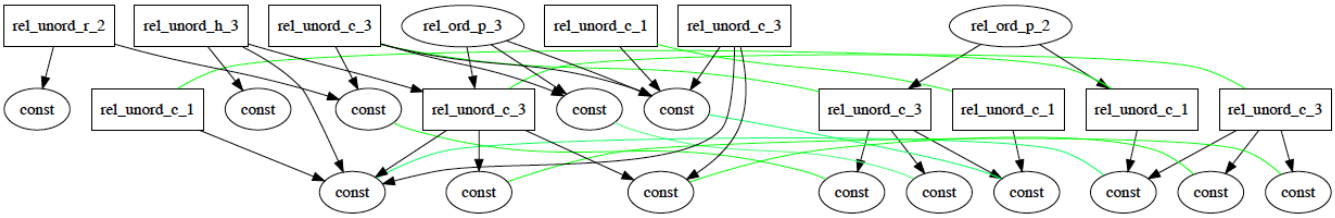

Figure 5: AMN output for an example from the Synthetic domain

consisted of base and target graphs, a set of correspondences, and a set of nodes from the base to be candidate inferences. Construction of synthetic examples begins with generating DAGs. Each DAG consists of a set of $k \in [2, 7]$ layers (with the particular k for a graph chosen at random). Each node is assigned an arity a, with the maximum arity being $a = 3$. Nodes at layer i can be connected to a nodes from lower layers (i.e., layer j with $j < i$) selected at random. Nodes with arity $a = 0$ are considered entities and nodes with non-zero arities (i.e., $a > 0$) are randomly assigned as predicates or functions and randomly designated as ordered or unordered.

To generate a training example, we first generate a set of random DAGs C, which will later become the correspondences. Next, we construct the base B by generating graphs *above* C. As each DAG is constructed in layers, this simply means that C is considered the lowest layers of B. Likewise, for the target T we build another set of graphs above C. The nodes of C are thus shared with both B and T. Each node of C is duplicated, producing one node for B and one node for T, and the resulting pair of nodes becomes a correspondence. Any element in B that was an ancestor of a node from C or a descendent of such an ancestor was considered a candidate inference. In Appendix 7.2 we provide a figure showing each component of a training example. During training, each generated example was turned into a batch of 8 inputs by repeatedly running the encoding procedure (which dynamically assigns node labels) over the original base and target.

5.2 Experimental Domains

Though training was done with synthetic data, we evaluated the effectiveness of AMN on both synthetic data and data used in previous analogy experiments. The corpus of previous analogy examples was taken from the public release of SME[1]. Importantly, AMN was *not* trained on the corpus of existing analogy examples (AMN never learned from a real-world analogy example). In fact, there was *no* overlap between the symbols (i.e., entities, functions, and predicates) used in that corpus and the symbols used for the synthetic data. We briefly describe each of the domains AMN was evaluated on below (detailed descriptions can be found in (Forbus et al. 2017)).

1. *Synthetic*: this domain consisted of 1000 examples generated with the same parameters as the training data (useful as a sanity check for AMN's performance).
2. *Visual Oddity*: this problem setting was initially proposed to explore cultural differences to geometric reasoning in (Dehaene et al. 2006). The work of (Lovett and Forbus 2011) modeled the findings of the original experiment computationally with qualitative visual representations and analogy. We extracted 3405 analogical comparisons from the computational experiment.
3. *Moral Decision Making*: this domain was taken from (Dehghani et al. 2008a), which introduced a computational model of moral decision making that used SME to reason through moral dilemmas. From the works of (Dehghani et al. 2008a,b), we extracted 420 analogical comparisons.
4. *Geometric Analogies*: this domain is from one of the first computational analogy experiments (Evans 1964). Each problem was an incomplete analogy of the form $A : B :: C : ?$, where each of A, B, and C were manually encoded geometric figures and the goal was to select the figure that best completed the analogy from an encoded set of possible answers. While in the original work all figures had to be manually encoded, in (Lovett et al. 2009; Lovett and Forbus 2012) it was shown that the analogy problems could be solved with structure-mapping over automatic encodings (produced by the CogSketch system (Forbus et al. 2011)). From that work we extracted 866 analogies.

5.3 Results and Discussion

Table 1a shows the results for AMN across different values of r, where r denotes the re-run hyperparameter detailed in Section 4.2. When evaluating on the synthetic data, the comparison set of correspondences was given by the data generator; whereas when evaluating on the three other analogy domains, the comparison set of correspondences was given by the output of SME. It is important to note that we are using SME as our stand-in for SMT (as it is the most widely accepted computational model of SMT). Thus, we do *not* want significantly different results from SME in the correspondence selection experiments (e.g., substantially higher or lower structural evaluation scores). Matching SME's performance (i.e., not producing higher or lower values) gives evidence that we are modeling SMT.

In the Struct. Perf. column, the numbers reflect the average across examples of the structural evaluation score of AMN divided by that of the comparison correspondence sets. For the other columns of Table 1a, the numbers represent average fractions of examples or correspondences (e.g., 0.684 should be interpreted as 68.4%). Candidate inference prediction performance was measured relative to the set of correspondences AMN generated, i.e., all candidate inferences were computed from the *predicted* correspondences, and treated as the true

[1] http://www.qrg.northwestern.edu/software/sme4/index.html

Domain	r	Struct. Perf.	Larger	Equiv.	Err. Free	1-to-1 Err.	PC Err.	Degen. Err.
Synthetic	1	0.713	0.000	0.313	0.346	0.007	0.102	0.020
Synthetic	16	0.952	0.001	0.683	0.695	0.005	0.020	0.011
Oddity	1	0.774	0.061	0.404	0.484	0.153	0.225	0.000
Oddity	16	0.955	0.074	0.485	0.564	0.131	0.139	0.000
Moral DM	1	0.610	0.014	0.021	0.093	0.002	0.170	0.030
Moral DM	16	0.958	0.081	0.164	0.329	0.000	0.041	0.016
Geometric	1	0.871	0.064	0.533	0.649	0.039	0.116	0.000
Geometric	16	1.040	0.069	0.714	0.788	0.029	0.043	0.000

(a) AMN correspondence prediction results for performance ratio (left), solution type rate (middle, ↑ better), and error rate (right, ↓ better)

Domain	r	Avg. CI F1	Avg. CI Prec.	Avg. CI Rec.	Avg. CI Acc.	Avg. CI Spec.
Synthetic	16	0.900	0.867	0.967	0.861	0.735
Oddity	16	0.992	0.995	0.994	0.991	0.911
Moral DM	16	0.899	0.834	0.985	0.832	0.439
Geometric	16	0.958	0.955	0.990	0.951	0.917

(b) AMN candidate inference prediction results

Table 1: AMN experimental results

positives. In many problems from the non-synthetic domains, every non-correspondence node was a candidate inference (which can lead to inflated precision and recall values). Thus, we also report the specificity (i.e., true negative rate) of AMN for *only* problems with non-candidate inference nodes.

In addition to our main results, we also provide qualitative examples of AMN's outputs on analogy problems and ablation studies for various aspects of AMN's design. Both the matching shown in Figure 5 as well as the solar system analogy shown in Figure 1 were produced by AMN. Further examples of AMN's outputs can be found in Appendix 7.5. Ablation experiments regarding the impact of both the signature graph and unit normalization of signature embeddings (each detailed in Section 4.1) are given in Appendix 7.3.

Analysis: The left side of Table 1a shows the average ratio of AMN's performance (labeled Struct. Perf.), as measured by structural evaluation score, against the comparison method's performance (i.e., data generator correspondences or SME). As can be seen, AMN produced matches with structural evaluation scores at 95-104% the level of SME on the non-synthetic domains, which indicates that it was finding similar structural matches. This is ideal as it shows that AMN matches SME's systematicity preference, and thus likely conforms fairly well to SMT in terms of systematicity.

The middle of Table 1a gives us the best sense of how well AMN modeled SMT. We observe AMN's performance in terms of the proportion of *larger*, *equivalent*, and *error-free* matches it produces (labeled Larger, Equiv., and Err. Free, respectively). Error-free matches do not contain degenerate correspondences or SMT constraint violations, whereas equivalent and larger matches are both error-free and have the same / larger structural evaluation score as compared to gold set of correspondences. The Equiv. column provides the best indication that AMN could model SMT. It shows that $\gtrapprox 50\%$ of AMN's outputs were SMT-satisfying, error-free analogical matches with the *exact same* structural score as SME (the lead computational model of SMT) in two of the non-synthetic analogy domains.

The right side of Table 1a shows the frequency of the different types of errors, including violations of the one-to-one / parallel connectivity constraints, and degenerate correspondences (labeled 1-to-1 Err., PC Err., and Degen. Err.). It shows that AMN had fairly low error rates across domains (except for Visual Oddity). Importantly, degenerate correspondences were very infrequent, which is significant because it verifies that AMN leveraged higher-order relational structure.

Table 1b shows that AMN was fairly effective in predicting candidate inferences. The high accuracy (labeled Avg. CI Acc.) scores for both the Visual Oddity and Geometric Analogies domains indicate that AMN was able to capture the notion of structural support when determining candidate inferences. The non-zero specificity (labeled Avg. CI Spec.) results show that, while it more often classified nodes as candidate inferences, it was capable of distinguishing non-candidate inference nodes as well.

6 Conclusions

In this paper, we introduced the Analogical Matching Network, a neural approach that learned to produce analogies consistent with Structure-Mapping Theory. AMN was trained on completely synthetic data and was capable of performing well on a varied set of analogies drawn from previous work involving analogical reasoning. AMN demonstrated renaming invariance, structural sensitivity, and the ability to find solutions in a combinatorial search space, all of which are key properties of symbolic reasoners and are known to be important to human reasoning.

References

Benaim, S.; Mokady, R.; Bermano, A.; and Wolf, L. 2020. Structural Analogy from a Single Image Pair. In *Computer Graphics Forum*. Wiley Online Library.

Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching word vectors with subword information. volume 5, 135–146. MIT Press.

Bowdle, B. F.; and Gentner, D. 1997. Informativity and asymmetry in comparisons. volume 34, 244–286. Elsevier.

Chalmers, D. J.; French, R. M.; and Hofstadter, D. R. 1992. High-level perception, representation, and analogy: A critique of artificial intelligence methodology. volume 4, 185–211. Taylor & Francis.

Chen, K.; Rabkina, I.; McLure, M. D.; and Forbus, K. D. 2019. Human-Like Sketch Object Recognition via Analogical Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1336–1343.

Clevert, D.-A.; Unterthiner, T.; and Hochreiter, S. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *International Conference on Learning Representations*.

Crouse, M.; Abdelaziz, I.; Cornelio, C.; Thost, V.; Wu, L.; Forbus, K.; and Fokoue, A. 2019. Improving Graph Neural Network Representations of Logical Formulae with Subgraph Pooling. In *arXiv preprint arXiv:1911.06904*.

Dehaene, S.; Izard, V.; Pica, P.; and Spelke, E. 2006. Core knowledge of geometry in an Amazonian indigene group. volume 311, 381–384. American Association for the Advancement of Science.

Dehghani, M.; Tomai, E.; Forbus, K.; Iliev, R.; and Klenk, M. 2008a. MoralDM: A Computational Modal of Moral Decision-Making. In *Proceedings of the Annual Meeting of the Cognitive Science Society*.

Dehghani, M.; Tomai, E.; Forbus, K. D.; and Klenk, M. 2008b. An Integrated Reasoning Approach to Moral Decision-Making. In *AAAI*, 1280–1286.

Eliasmith, C.; and Thagard, P. 2001. Integrating structure and meaning: A distributed model of analogical mapping. volume 25, 245–286. Wiley Online Library.

Emami, P.; and Ranka, S. 2018. Learning permutations with sinkhorn policy gradient. In *arXiv preprint arXiv:1805.07010*.

Evans, T. G. 1964. A program for the solution of a class of geometric-analogy intelligence-test questions. Technical report, AIR FORCE CAMBRIDGE RESEARCH LABS LG HANSCOM FIELD MASS.

Falkenhainer, B.; Forbus, K. D.; and Gentner, D. 1989. The structure-mapping engine: Algorithm and examples. volume 41, 1–63.

Forbus, K.; Usher, J.; Lovett, A.; Lockwood, K.; and Wetzel, J. 2011. CogSketch: Sketch understanding for cognitive science research and for education. volume 3, 648–666. Wiley Online Library.

Forbus, K. D.; Ferguson, R. W.; Lovett, A.; and Gentner, D. 2017. Extending SME to handle large-scale cognitive modeling. volume 41, 1152–1201. Wiley Online Library.

Gentner, D. 1983. Structure-mapping: A theoretical framework for analogy. volume 7, 155–170. Elsevier.

Gentner, D. 2003. Why we're so smart. In *Language in mind: Advances in the study of language and thought*, volume 195235.

Gentner, D.; Brem, S.; Ferguson, R. W.; Markman, A. B.; Levidow, B. B.; Wolff, P.; and Forbus, K. D. 1997. Analogical reasoning and conceptual change: A case study of Johannes Kepler. volume 6, 3–40. Taylor & Francis.

Gentner, D.; and Markman, A. B. 1993. Analogy–Watershed or Waterloo? Structural alignment and the development of connectionist models of analogy. In *Advances in Neural Information Processing Systems*, 855–862.

Gentner, D.; and Markman, A. B. 1998. Analogy-based reasoning. In *The handbook of brain theory and neural networks*, 91–93. MIT Press.

Georgiev, D.; and Lió, P. 2020. Neural Bipartite Matching. In *arXiv preprint arXiv:2005.11304*.

Gick, M. L.; and Holyoak, K. J. 1983. Schema induction and analogical transfer. Elsevier.

Halford, G. S.; Wilson, W. H.; Guo, J.; Gayler, R. W.; Wiles, J.; and Stewart, J. 1994. Connectionist implications for processing capacity limitations in analogies. In *Advances in Connectionist and Neural Computation Theory*.

Hill, F.; Santoro, A.; Barrett, D. G.; Morcos, A. S.; and Lillicrap, T. 2019. Learning to make analogies by contrasting abstract relational structure. In *International Conference on Learning Representations*.

Hofstadter, D. 1995. *Fluid concepts and creative analogies: Computer models of the fundamental mechanisms of thought.* Basic books.

Hofstadter, D. R. 2001. Analogy as the core of cognition. 499–538. Keith J. Holyoak, and Boicho N. Kokinov. Cambridge MA: The MIT Press.

Holyoak, K. J.; Holyoak, K. J.; and Thagard, P. 1995. *Mental leaps: Analogy in creative thought*. MIT press.

Holyoak, K. J.; Junn, E. N.; and Billman, D. O. 1984. Development of analogical problem-solving skill. 2042–2055. JSTOR.

Holyoak, K. J.; and Thagard, P. 1989. Analogical mapping by constraint satisfaction. volume 13, 295–355. Wiley Online Library.

Hummel, J. E.; and Holyoak, K. J. 1997. Distributed representations of structure: A theory of analogical access and mapping. volume 104, 427. American Psychological Association.

Hummel, J. E.; and Holyoak, K. J. 2005. Relational reasoning in a neurally plausible cognitive architecture: An overview of the LISA project. volume 14, 153–157. SAGE Publications Sage CA: Los Angeles, CA.

Kahneman, D.; Treisman, A.; and Gibbs, B. J. 1992. The reviewing of object files: Object-specific integration of information. volume 24, 175–219. Elsevier.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. In *arXiv preprint arXiv:1412.6980*.

Klenk, M.; Forbus, K. D.; Tomai, E.; Kim, H.; and Kyckelhahn, B. 2005. Solving everyday physical reasoning problems by analogy using sketches. In *AAAI Conference on Artificial Intelligence*.

Kool, W.; Van Hoof, H.; and Welling, M. 2018. Attention, learn to solve routing problems! In *International Conference on Learning Representations*.

Lovett, A.; and Forbus, K. 2011. Cultural commonalities and differences in spatial problem-solving: A computational analysis. volume 121, 281–287. Elsevier.

Lovett, A.; and Forbus, K. 2012. Modeling multiple strategies for solving geometric analogy problems. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 34.

Lovett, A.; Tomai, E.; Forbus, K.; and Usher, J. 2009. Solving geometric analogy problems through two-stage analogical mapping. volume 33, 1192–1231. Wiley Online Library.

Lu, H.; Liu, Q.; Ichien, N.; Yuille, A. L.; and Holyoak, K. J. 2019. Seeing the Meaning: Vision Meets Semantics in Solving Pictorial Analogy Problems. In *CogSci*, 2201–2207.

Mikolov, T.; Yih, W.-t.; and Zweig, G. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 746–751.

Mitchell, M. 1993. Analogy-making as perception - a computer model. In *Neural network modeling and connectionism*.

Novick, L. R. 1988. Analogical transfer, problem similarity, and expertise. volume 14, 510. American Psychological Association.

O'Donoghue, T. V. D.; and Keane, M. 1999. Computability as a limiting cognitive constraint: Complexity concerns in metaphor comprehension about which cognitive linguists should be aware. In *Cultural, Psychological and Typological Issues in Cognitive Linguistics: Selected papers of the bi-annual ICLA meeting in Albuquerque, July 1995*, volume 152, 129. John Benjamins Publishing.

Reed, S. E.; Zhang, Y.; Zhang, Y.; and Lee, H. 2015. Deep visual analogy-making. In *Advances in Neural Information Processing Systems*, 1252–1260.

Ribeiro, D.; Hinrichs, T.; Crouse, M.; Forbus, K.; Chang, M.; and Witbrock, M. 2019. Predicting State Changes in Procedural Text using Analogical Question Answering. In *Advances in Cognitive Systems*.

Sagi, E.; Gentner, D.; and Lovett, A. 2012. What difference reveals about similarity. volume 36, 1019–1050. Wiley Online Library.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, 5998–6008.

Veale, T.; and Keane, M. T. 1997. The competence of sub-optimal theories of structure mapping on hard analogies. In *IJCAI*, 232–237.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, 2692–2700.

Wang, R.; Yan, J.; and Yang, X. 2019. Learning combinatorial embedding networks for deep graph matching. In *Proceedings of the IEEE International Conference on Computer Vision*, 3056–3065.

Wetzel, L. 2009. *Types and tokens: On abstract objects*. MIT Press.

Zhang, C.; Gao, F.; Jia, B.; Zhu, Y.; and Zhu, S.-C. 2019. Raven: A dataset for relational and analogical visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5317–5327.

Zhou, L.; Cui, P.; Yang, S.; Zhu, W.; and Tian, Q. 2019. Learning to learn image classifiers with visual analogy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 11497–11506.